Bulletin T.CXLV de l'Académie serbe des sciences et des arts − 2013
Classe des Sciences mathématiques et naturelles
Sciences mathématiques, № 38

NETWORK ALIGNMENT USING SELF-RETURNING WALKS

D. CVETKOVIĆ, [1] IRENA M. JOVANOVIĆ [2]

(Presented at the 4th Meeting, hold on May 31, 2013)

A b s t r a c t. We propose a new approach to the network alignment problem. We define the measure of similarity between vertices of considered networks using the numbers of self-returning walks at particular vertices. These numbers are related to graph invariants called graph angles which are known in spectral graph theory. We indicate advantages of our approach in comparison with existing procedures for network alignment.

AMS Mathematics Subject Classification (2000): 05C50, 05C85, 68R10
Key Words: graph spectra, network alignment, graph algorithms

1. *Introduction*

An *undirected graph* G is a mathematical structure that can be defined as an ordered pair $G = (V, E)$, where elements of the set V are called *vertices* of G, while the elements of the set E are two-element subsets of V and they

[1]The work is supported by Serbian Ministry of Education, Science and Technological Development, Projects 174033 and F-159.
[2]The work is supported by Serbian Ministry of Education, Science and Technological Development, Projects 174033 and III 45003.

are called *edges*. If a real number, denoted as a *weight*, is associated with each edge $e \in E$, then G is called a *weighted graph*. Graphs with large number of vertices are usually called *networks*.

Networks can be used to model a wide range of real-world relations, processes and phenomena in various research domains, such as computer science, linguistics, chemistry and physics, sociology and biology. In computer science, graphs are used to represent networks of communication, data organization, computational devices, etc. A graph is an evident model for a molecule, where vertices represent atoms and edges bonds. As a consequence of the usage of the graph theory in sociology, many different types of *social network graphs* are developed. On that way, the *friendship graphs* represent whether given number of people know each other, while the *collaboration graphs* model whether two people cooperate.

Analyzing the structure of the biological networks can provide insights into evolution, protein function, cells function and protein-protein interaction, which will imply prediction of disease progress, synthesis of new medicaments and better understanding of biology and evolution, as well. Numerous biological theoretical and experimental studies have led to various types of biological networks: *gene regulatory networks*, *metabolic networks*, *signaling networks*, *neuronal networks*, etc. The most intensely analyzed are the *protein-protein interaction (PPI) networks*, that are modelled as an undirected unweighted graph, whose vertex set represents the set of proteins, while the edge set represents the interactions between them.

Comparative analysis of, specially biological, networks are one of the foremost challenges for today researchers. Nevertheless, comparison of large graphs is computationally infeasible, because it means to solving the *subgraph isomorphism problem*: if one graph exists as an exact subgraph of another graph? This problem is NP-complete, so there is a need to develop heuristic strategies for its approximate solving. *Network alignment* is one of them.

The main task in network alignment is to detect one or multiple possible mappings between the vertices of the comparative networks. These mappings need not be defined for all vertices in the network. By network alignment one aims to find as largest as possible a common subgraph between the two networks. The edges of this subgraph correspond to the conserved edges implied by defined mappings. We distinguish two types of network alignment algorithms depends on the nature of mappings - local and global network alignment algorithms. While *local alignments* detect local regions of similarity between the networks, *global network alignment algorithms* match every vertex in the one of the networks to some vertex in the other network or mark some of the vertices as "gap nodes" (i.e. with no match in the other network) (see [17]). That is, a global network alignment algorithm means the presence of a single mapping across all parts of the input network. So, a global alignment of two networks, $G(V_G, E_G)$ and $H(V_H, E_H)$, such that $|V_G| \leq |V_H|$, is the set of ordered pairs (i, j), $i \in V_G$ and $j \in V_H$, such that no two ordered pairs have the common vertex. Each such ordered pair is called an *aligned pair* (see [13] or [14]).

Numerous algorithms for global network alignment have been developed and mostly of them have been implemented for aligning of PPI networks: the first between them was IsoRank (see [17]) that has been extended to perform local and global alignments between multiple networks (see [12]), PATH and GA algorithms (see [20]), next algorithms that can align any type of network not only biological one as GRAAL (see [10]) and H-GRAAL (see [14]), followed by MI-GRAAL (see [11]) that can use any number and type of vertex similarity measures and C-GRAAL (see [13]) whose measure depends only of the underlying network topology, etc. In general case, all of these algorithms have been realized in two phases. In the first one, the measure of similarity between vertices of G and vertices of H have been computed, and in the second the mappings between vertices, i.e. the aligned pairs, are constructed related to the computed vertex similarity. The measure of similarity is always defined taking into account the neighbourhoods of the corresponding vertices and mainly is not spectrally based, except, for example, for the case of IsoRank algorithm. Although by global network alignment we seek to find a solution among all possible global matchings, we, in fact, need to find a maximal matching in a weighted bipartite graph on vertex sets of the graphs G and H whose edge weights are defined measures of similarity between vertices of graphs G and H, respectively.

As proposed in [3] and [8], a new measure of similarity between vertex i of G and vertex j of H should be based on the difference $\mathcal{H}_i^G(t) - \mathcal{H}_j^H(t)$ of generating functions for the numbers of self-returning walks for these vertices. On that way, this spectrally characterized measure will also involve the vertex neighbourhood, which will be extended to the whole graph unlike the measures where the neighbourhood is very limited (see, for example, [10]). In this paper, we shall point out the main advantages of this approach and compare it with the existing algorithms for global network alignment by theoretical tools.

The rest of the paper is organized as follows. Section 2 gives some definitions and results on walks in graphs. In Section 3 we present necessary material on graph angles which is relevant for our approach to network alignment. Section 4 reviews existing algorithms for the network alignment problem. Our approach to the network alignment problem is described in Section 5. Sections 6 and 7 are devoted to complexity analysis and concluding remarks.

2. *Walks in graphs*

Let G be a graph without loops and with the vertex set $V = \{1, 2, \dots, n\}$. In this note, we are concerned with the number of self-returning walks of the graph G at its vertex j. Remember, by a *walk* of length k in G we mean any sequence of (not necessarily distinct) vertices $j_1, j_2, \dots, j_k, j_{k+1} \in \{1, 2, \dots, n\}$ such that for each $i = 1, 2, \dots, k$ there is an edge from j_i to j_{i+1}. A walk starting and terminating at the vertex $j \in \{1, 2, \dots, n\}$ is called a *closed walk* or a *self-returning walk* at j. A self-returning walk in a graph is called a *spanning self-returning walk* if it is passes through all vertices of the graph. A new spectrally based formula for counting the number of spanning self-returning walks of certain length in a rooted graph (i.e. a graph with a distinguished vertex) is proposed in [9].

Counting walks with specified properties in a graph (or digraph) is related to graph spectra by the following well-known result (see [4] p. 44).

Theorem 1. *If A is the adjacency matrix of a graph, then the (i, j)-entry $a_{ij}^{(k)}$ of the matrix A^k is equal to the number of walks of length k that originate at vertex i and terminate at vertex j.*

The normalized positive eigenvector belonging to the largest A-eigenvalue of a connected graph is called the *principal eigenvector*.

Coordinates of the principal eigenvector are related to vertex neighbourhoods because they are asymptotically proportional to the number of walks of length k starting at particular vertices (*out-going walks*). The following relevant theorem of T.H. Wei [19] is noted in [5], p. 26:

Theorem 2. *Let $N_k(i)$ be the number of walks of length k starting at vertex i of a non-bipartite connected graph G with vertices $1, 2, \dots, n$. Let $s_k(i) = N_k(i) \cdot \left(\sum_{j=1}^{n} N_k(j)\right)^{-1}$. Then, for $k \to \infty$, the vector $(s_k(1), s_k(2), \dots, s_k(n))^T$ tends towards the principal eigenvector of G.*

3. *Graph angles*

Since eigenvectors are not graph invariants it is reasonable to extend eigenvalue based techniques by some invariants of the eigenspaces called *graph angles.*

Let G be a graph on n vertices with distinct eigenvalues $\mu_1, \mu_2, \ldots, \mu_m$ ($\mu_1 > \mu_2 > \cdots > \mu_m$) and let $S_1, S_2, \ldots, S_m$ be the corresponding eigenspaces. Let $\{e_1, e_2, \ldots, e_n\}$ be the standard (orthonormal) basis of R^n. The numbers $\alpha_{pq} = \cos \beta_{pq}$ ($p = 1, 2, \ldots, m; q = 1, 2, \ldots, n$), where β_{pq} is the angle between S_p and e_q, are called *graph angles.* The sequence α_{pq} ($q = 1, 2, \ldots, n$) is called the *eigenvalue angle sequence* corresponding to the eigenvalue μ_p ($p = 1, 2, \ldots, m$). We also define the *angle matrix* of G, i.e. an $m \times n$ matrix (m is the number of its distinct eigenvalues, while n is the order of G) as a matrix (α_{ij}). This matrix is a graph invariant if its columns are ordered lexicographically. The rows of the angle matrix are called the *standard eigenvalue angle sequences.*

Let $x_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ ($i = 1, 2, \ldots, n$) be orthonormal eigenvectors of G. Define $M_p = \{j \mid Ax_j = \mu_p x_j\}$. We have $\alpha_{pq}^2 = \sum\limits_{j \in M_p} x_{jq}^2$ for squares of angles of G. This formula holds for any choice of orthonormal eigenvectors of G (cf. [5], p. 76).

An overview of results on graph angles is given in [5] including the characterizing properties of graph angles.

Let A be the adjacency matrix of a graph G , and let $N_k(j) = a_{jj}^{(k)}$ be the number of walks of length k in G originating and terminating at vertex j. The following formula follows from the *spectral decomposition* of the adjacency matrix A (cf. [5], p. 81):

$$N_s(j) = a_{jj}^{(s)} = \sum_{i=1}^{n} \mu_i^s \alpha_{ij}^2.$$

If $\mathcal{H}_j(t)$ denotes the generating function $\sum_{k=0}^{\infty} N_k(j) t^k$, using the previous formula, we can obtain (see [5], p. 82):

$$\mathcal{H}_j(t) = \sum_{k=0}^{\infty} N_k(j) t^k = \sum_{k=0}^{\infty} t^k \sum_{i=1}^{m} \alpha_{ij}^2 \mu_i^k = \sum_{i=1}^{m} \frac{\alpha_{ij}^2}{1 - \mu_i t}.$$

On the other hand, we have $\mathcal{H}_j(t) = 1 + d_j t^2 + 2t_j t^3 + \cdots$, where d_j is the degree of vertex j and t_j is the number of triangles containing j. The quantity t_j is also called the *clustering coefficient* of the vertex j.

The degree d_j of the vertex j, and the number t_j of triangles containing the vertex j, can also be represented using graph angles:

$$d_j = \sum_{i=1}^{m} \alpha_{ij}^2 \mu_i^2, \qquad t_j = \frac{1}{2} \sum_{i=1}^{m} \alpha_{ij}^2 \mu_i^3.$$

Similar formulas that use graph angles describe the number of quadrangles and pentagons in a graph, and can be also found in [5], p. 82.

If $P_G(\lambda) = \det(\lambda I - A)$ is the characteristic polynomial of the graph G, then the generating function can be obtained by the formula

$$\mathcal{H}_j^G(t) = P_{G-j}(\frac{1}{t})/tP_G(\frac{1}{t}),$$

since

$$P_{G-j}(x) = P_G(x) \sum_{i=1}^{m} \frac{\alpha_{ij}^2}{x - \mu_i}$$

holds for the characteristic polynomial of vertex-deleted subgraph of a graph.

4. *Algorithms for the network alignment problem*

As we have seen, detecting similarities between networks is frequently called *alignment* of networks. The general idea is realized in two phases in the following way.

In the first phase, for the two given graphs $G = (V_G, E_G)$ and $H = (V_H, E_H)$, a measure of similarity between vertices of G and vertices of H is introduced by some definition, that always takes into account the neighbourhoods of the corresponding vertices. On that way, for example, vertices of the same degree should be considered as being more similar than those with different degrees. Let $R_{i,j}$ be the measure of similarity between vertex i of G and vertex j of H. Let B be the bipartite graph on vertex sets of the graphs G and H with edge weights $R_{i,j}$. Via the second phase of optimal alignment, we want to find a maximal matching with a maximal sum of weights in B. This matching defines subgraphs of graphs G and H which are similar w.r.t. the introduced similarity between vertices of G and vertices of H. Finding a maximal matching with a maximal sum of weights in a bipartite graph can efficiently be performed by existing algorithms for the assignment problem in combinatorial optimization (see, for example, [16]), among whom the Hungarian algorithm is mostly used. In contrast to the exposed technique, some algorithms are based on the so called "seed-and-extend" approach in the second phase.

The first algorithm for global network alignment is called IsoRank. The computation of the optimal global alignment between two networks is based on a heuristic that vertices i and j should be matched if their neighbors can also be well matched (see [17]). This measure has the property that for any pair (i, j) it is equal to the mean value of similarities between all pairs (u, v) where u is a neighbour of i and v is a neighbour of j. That is represented by the following formula:

$$R_{ij} = \sum_{u \in N(i)} \sum_{v \in N(j)} \frac{1}{|N(u)||N(v)|} R_{uv},$$

for $i \in V_G$ and $j \in V_H$, and where $N(a)$ is the set of neighbors of vertex a.

The algorithm uses graph eigenvectors, and is similar to the algorithm PageRank, used in the Internet search (see [2]). Here the measure of similarity $R_{i,j}$ is spectrally based because the vector of $R_{i,j}$, for all $i \in V_G$ and $j \in V_H$, is the principal eigenvector of $|V_G||V_H| \times |V_G||V_H|$ matrix A defined as follows: $A[i, j][u, v] = \frac{1}{|N(u)||N(v)|}$, if $\{i, u\} \in E_G$ and $\{j, v\} \in E_H$, and $A[i, j][u, v] = 0$, otherwise.

By Theorem 2 this approach is related to enumeration of long out-going walks of particular vertices.

The principal eigenvector of a graph whose adjacency matrix is very large can efficiently be computed by an iterative algorithm called the *power method* (see, for example, [6]).

A similar concept of the measure of similarity has been introduced in [1], even more generally between vertices of two digraphs and applied to synonym extraction from a dictionary. When considering undirected graphs, we obtain the construction used in IsoRank.

After the principal eigenvector has been computed, the vertex mappings, i.e. the aligned pairs, are detected by solving the maximum - weight matching problem in the bipartite graph, as we described previously.

The same approach as in the second phase of the IsoRank algorithm, i.e. finding the aligned pairs by solving maximum (or minimum)-weight bipartite matching problem that includes optimal matching algorithms, as Hungarian algorithm is, has been implemented in the algorithm called H-GRAAL (see [14]) and also used as one of the strategies for solving node matching problem between networks (see [18]).

The *node matching problem* is essentially the same as the network alignment problem. In both of them the main task is to find the pairs of matched vertices from compared networks, respectively, i.e. the aligned pairs, but in the node matching problem not only by using the structural or topological data of networks under study yet by using the *revealed matched nodes*, i.e. the pairs of vertices that have been already revealed for matching. The set of the revealed matched nodes consists of the pairs of vertices whose the first component represents selected vertex with larger degree in the reference network, while the second component of the named pair is chosen in the other network by some dedicated methods in dependence of the nature of considered network. The measure of similarity of each pair of the remaining vertices, i.e. those different from revealed matched nodes, has been calculated based on the number of pairs of revealed matched nodes around that considered pair. In [18] for this calculation the Jaccard index has been adopted (see [7]).

In its second phase H-GRAAL algorithm uses the Hungarian algorithm for finding maximum-weight bipartite matching with respect to the measure of similarity between vertices that, again, is not spectrally characterized. Namely, the cost function in this approach is based on a *signature similarity* between vertices of compared networks. It means that a vertex is characterized by a 73-dimensional vector of *graphlet degrees*, which describes the topology of its neighborhood.

A graph G with a distinguished vertex j is called a *rooted graph*. The vertex j is called the *root* of G and G is said to be *rooted at* j. A connected rooted graph is also called a *graphlet*. More precisely, a graphlet in a graph is a connected induced rooted subgraph of a graph (see [10]). Suppose that among graphlets there are g mutually non-isomorphic ones and suppose that non-isomorphic graphlets are indexed by integers $0, 1, \dots, g-1$ in an arbitrary way. The graphlet indexed k is called the *graphlet of type* k. Both algorithms H-GRAAL (see [14]), and also GRAAL (see [10]), deals with the graphlets up to five vertices. There are 30 connected graphs on 2 to 5 vertices with 73 different roots, i.e. 73 different connected rooted graphs (i.e. graphlets) on 2 to 5 vertices. More precisely, 73 types of graphlets are used in these two algorithms. In that case, the index k corresponds to one of the 73 roots that are numerated with numbers 0,1,. . . ,72 (see the Figure 1). Let u_k^i be the number of graphlets of type k at vertex i. On that way, the k-th coordinate, u_k^i, of the mentioned 73-dimensional vector represents the number of graphlets of type k (graphlet degree) in which the considered vertex i appears. This vector is called the *signature* of vertex i.

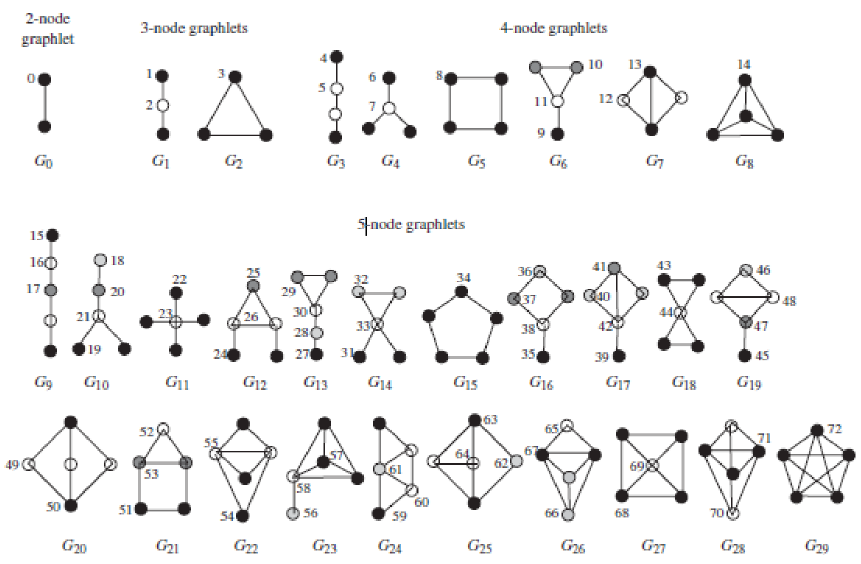

Fig. 1. All graphlets up to five vertices with the labelled roots (from [10])

The weight that is assigned to the edge $\{i, j\}$, for every pair of vertices $i \in V_G$ and $j \in V_H$ in the bipartite graph, for which in its second phase H-GRAAL finds the minimum weight matching, is represented by the following cost function:

$$C(i, j) = 2 - ((1 - \alpha)T(i, j) + \alpha S(i, j)).$$

In the exposed formula,

$$T(i, j) = \frac{d_i + d_j}{\max_{i \in V_G} d_i + \max_{j \in V_H} d_j},$$

where d_i stands for the degree of the vertex i and $\max_{i \in V_G} d_i$ for the maximum vertex degree in the graph G (and respectively H); $S(i, j)$ is the signature similarity of vertices i and j, while a parameter $\alpha \in [0, 1]$ controls the contribution of vertex signature similarity between vertices i and j.

Here and in the next section we shall assume that vertex sets V_G and V_H of graphs G and H are disjoint so that all quantities indexed by i, j are related to graphs G and H respectively.

The signature similarity $S(i, j)$ between vertices i and j can be computed as: $S(i, j) = 1 - D(i, j)$, where $D(i, j)$ is the *total distance* between vertices i and j, and is given by the following formula:

$$D(i,j) = \frac{\sum_{k=0}^{72} D_k(i,j)}{\sum_{k=0}^{72} w_k}.$$

Here, w_k is the *weight of the graphlet of type* k and is defined as (see [15]):

$$w_k = 1 - \frac{\log(o_k)}{\log(73)}.$$

The integer o_k counts the number of connected induced subgraphs of the graphlet of type k rooted at the same vertex, including the graphlet itself. Thus, for example, $o_{16} = 6$, because the graphlets of types 0, 1, 2, 4, 5, 16 are the connected induced subgraphs of the graphlet of type 16, that contain the same root. The computational results show that the values of w_k belong to the interval $[0.488, 0.838]$ except for $w_0 = 1$. We have $w_{53} = w_{60} = 0.488$ and $w_1 = w_2 = w_3 = 0.838$. If $w = \sum_{k=0}^{72} w_k$, from the previous formula we can easily obtain that $w \approx 45.135$.

The distance $D_k(i,j)$ between the k-th coordinates of the 73-dimensional vectors of graphlet degrees assigned to the vertices i and j is designated by:

$$D_k(i,j) = w_k \frac{|\log(u_k^i + 1) - \log(u_k^j + 1)|}{\log(max(u_k^i, u_k^j) + 2)}.$$

If we introduce the function $LD(x,y) = \frac{|\log(x+1) - \log(y+1)|}{\log(max(x,y)+2)}$, the formula for the total distance becomes:

$$\begin{aligned} D(i,j) = \;& \tfrac{1}{w} \textstyle\sum_{k=0}^{72} D_k(i,j) = 0.022 \sum_{k=0}^{72} D_k(i,j) = \\ & 0.022 \times (w_0\, LD(u_0^i, u_0^j) + w_1\, LD(u_1^i, u_1^j) + w_2\, LD(u_2^i, u_2^j) \\ & + w_3\, LD(u_3^i, u_3^j) + \cdots) = \\ & 0.022 \times (LD(d_i, d_j) + 0.838\, LD(u_1^i, u_1^j) + 0.838\, LD(\tbinom{d_i}{2} \\ & - t_i, \tbinom{d_j}{2} - t_j) + 0.838\, LD(t_i, t_j) + 0.744\, LD(u_4^i, u_4^j) + \cdots) = \\ & 0.022\, LD(d_1, d_j) + 0.018\, LD(u_1^i, u_1^j) + 0.018\, LD(\tbinom{d_i}{2} - t_i, \tbinom{d_j}{2} - t_j) \\ & + 0.018\, LD(t_i, t_j) + 0.016\, LD(u_4^i, u_4^j) + \cdots, \end{aligned}$$

where d_l and t_l denote, as before, the degree of the vertex l and the number of triangles containing the vertex l, respectively. All presented numerical values are approximate.

Our feeling is that the formula for $D(i,j)$ is a bit artificially constructed. It is not clear when $D(i,j) = 1$ although it is in the interval $[0,1]$. It could happen that for all pairs i, j it is much below 1. Also the weights w_k are in a narrow interval so that the influence of particular graphlet frequencies seems not to be well balanced.

The paper [14] also suggests application of a dynamic version of the Hungarian algorithm. Perturbation of a small number of edge weights in the original problem, for which the optimal matching is already known, will imply a new variant of a problem which recalls to the strategy with revealed matched nodes and can be performed for $O(n^2)$ time. Besides, the paper [10] underlines the possibility of adding protein sequence component to the cost function, which imply eventually better alignment of the PPI networks. However, one of the mostly important advantages of GRAAL is that there is no need to implement such kind of information, so it can align any two networks, not only biological.

While H-GRAAL and GRAAL practically coincide in the first phase, their second phase is quite different. H-GRAAL gives an optimal network alignment during GRAAL stands for the greedy seed-and-extend approach (see [10]). It means that the algorithm at the beginning of the second phase chooses a pair of vertices from the networks under study with high graphlet degree similarity. This pair is designated as the initial seed. Then, it greedily expands the alignment radially outward around the seed as far as possible.

The algorithm GRAAL has been upgraded into the advanced global alignment algorithms that are all equal in the second phase i.e. that have seed-and-extend nature. One of them, MI-GRAAL algorithm (i.e. Matching-based Integrative GRAph ALigner, see [11]) represents the improvement related to GRAAL in the first phase. Namely, it has a possibility to integrate any number and type of similarity measures between network vertices, that as a consequence has the *stable alignments* (i.e. the alignments that are very similar for almost all runs of the algorithm). C-GRAAL algorithm (Common-neighbors based GRAph ALigner, see [13]) in choosing a seed uses a measure that is denoted as *combined neighbour density* and is based solely on the underlying network topology. A larger combined neighbour density between two vertices means a higher topological similarity between their extended neighborhoods out to distance 2, in comparison to the strategy with graphlets where distance is 4.

All previously mentioned algorithms are summarized in the following table according to the similarities in one of the two working phases.

Table 1. Network alignment algorithms

phase 1	phase 2	
	optimal matching	seed-and-extend matching
H-GRAAL	IsoRank	GRAAL
	H-GRAAL	MI-GRAAL
	Optimal node matching algorithm from [18]	C-GRAAL
GRAAL		

5. *A new approach*

We suggest a new approach to the network alignment algorithms in their first phase. Namely, we think that in problems of network alignment vertices should be characterized by generating functions $\mathcal{H}_j(t)$ for the number of self-returning walks at vertex j. This function depends on the vertex neighbourhood which is in this case extended to the whole graph unlike the method with graphlets realized in GRAAL and H-GRAAL, where the neighbourhood is very limited.

For example, the measure of similarity between vertices i and j can be defined in some way using the difference $\mathcal{H}_i^G(t) - \mathcal{H}_j^H(t)$ of generating functions of vertex i in G and vertex j in H. In particular, we can use the formula:

$$d(i,j) = A|\mathcal{H}_i^G(t_0) - \mathcal{H}_j^H(t_0)|$$

for a sufficiently small positive value t_0 with A chosen in such a way that the maximal value of $d(i,j)$ over all pairs (i,j) is equal to 1. Note that the choice of A depends on t_0. In this way the distance 1 is always achieved while in the approach by graphlets it is not always the case.

The value t_0 can be chosen in the interval $(0,R)$, where R is the radius of convergence of the power series:

$$\mathcal{H}_i^G(t) - \mathcal{H}_j^H(t) = \sum_{k=0}^{+\infty}(N_k^G(i) - N_k^H(j))t^k$$

$$= \sum_{k=0}^{+\infty}\left(\sum_{p=1}^{m_G}(\mu_p^G)^k(\alpha_{pi}^G)^2 - \sum_{q=1}^{m_H}(\mu_q^H)^k(\alpha_{qj}^H)^2\right),$$

where $N_k^G(s)$ ($N_k^H(s)$) is the number of self-returning walks of length k at vertex s in G (H), m_G and m_H denote the number of distinct eigenvalues of graphs G and H, respectively, while μ_p^G and α_{pi}^G, for $p = 1, 2, \ldots, m_G$ and $i = 1, 2, \ldots, |V(G)|$, as before, stand for the distinct eigenvalues and angles of graph G (and analogously for graph H). One can easily conclude that if $\mu_1^G > \mu_1^H$, than $R = \frac{1}{\mu_1^G}$, and analogously, if $\mu_1^G < \mu_1^H$, the radius of convergence will be $R = \frac{1}{\mu_1^H}$. In the case when $\mu_1^G = \mu_1^H$, the radius of convergence is $R = \frac{1}{\mu_1^G}$ if $\alpha_{1i}^G \neq \alpha_{1j}^H$, while in contrary, the value of the radius of convergence will depend from the relation of the second largest distinct eigenvalues of graphs G and H, similarly as in the previous case. So, we can assume that the value t_0 belongs to the interval $\left(0, \frac{1}{\max\{\mu_1^G, \mu_1^H\}}\right)$.

The following formulas for the number od self-returning walks of certain length at a particular vertex j of a graph F have been proved in [9]:

$N_2(j) = d_j;$

$N_3(j) = 2t_j;$

$N_4(j) = u_0 + u_1 + 2u_2 + 4u_3 + 2q_j;$

$N_5(j) = 10u_3 + 2u_9 + 2u_{10} + 4u_{11} + 8u_{12} + 12u_{13} + 30u_{14} + 2p_j;$

$N_6(j) = u_0+3u_1+6u_2+20u_3+u_4+2u_5+2u_6+6u_7+18u_8+4u_9+5u_{10}+10u_{11}+ 30u_{12}+42u_{13}+120u_{14}+2u_{35}+2u_{36}+2u_{37}+4u_{38}+2u_{39}+2u_{40}+2u_{41}+4u_{42}+ 4u_{43}+8u_{44}+2u_{45}+2u_{46}+4u_{47}+2u_{48}+12u_{49}+18u_{50}+4u_{51}+4u_{52}+6u_{53}+ 12u_{54}+18u_{55}+6u_{56}+6u_{57}+12u_{58}+12u_{59}+14u_{60}+20u_{61}+16u_{62}+22u_{63}+ 18u_{64}+32u_{65}+34u_{66}+46u_{67}+46u_{68}+56u_{69}+90u_{70}+108u_{71}+216u_{72}+2h_j,$

where d_j, t_j, q_j, p_j and h_j denote the degree of vertex j, the number of triangles containing the vertex j, the number of quadrangles containing the vertex j, the number of pentagons containing the vertex j and the number of hexagons containing the vertex j, respectively, while u_i are the coordinates of the signature of the vertex j.

We shall apply these formulas to our graphs G and H and then the value of j will determine to which of these two graphs all mentioned quantities are related. Quantities u_s will have a superscript i or j.

Using these formulas our formula for the distance between vertices i and j becomes:

$$d(i,j) = A\,|\mathcal{H}_i^G(t_0) - \mathcal{H}_j^H(t_0)| =$$

$$A\,|\textstyle\sum_{k=0}^{+\infty}(N_k^G(i) - N_k^H(j))\,t_0^k| =$$

$$A\,|(N_2^G(i) - N_2^H(j))\,t_0^2 + (N_3^G(i) - N_3^H(j))\,t_0^3 + (N_4^G(i) - N_4^H(j))\,t_0^4$$

$$+(N_5^G(i) - N_5^H(j))\,t_0^5 + \cdots| =$$

$$A\,t_0^2\,|(d_i - d_j)\; + 2(t_i - t_j)\,t_0 + ((d_i - d_j) + (u_1^i - u_1^j) + 2(u_2^i - u_2^j) + 4(t_i - t_j) +$$

$$+2(q_i - q_j))\,t_0^2 + (10\,(t_i - t_j) + 2\,(u_9^i - u_9^j) + 2\,(u_{10}^i - u_{10}^j) + 4\,(u_{11}^i - u_{11}^j) +$$

$$+8\,(u_{12}^i - u_{12}^j) + 12\,(u_{13}^i - u_{13}^j) + 30\,(u_{14}^i - u_{14}^j) + 2\,(p_i - p_j))\,t_0^3 + \cdots|. \tag{1}$$

Since the value of t_0 practically is very small, the summands from formula (1) containing t_0^5 and higher powers of t_0 can be ignored, so we can take:

$$\begin{aligned} d(i,j) \approx \;\; & A\,t_0^2\,|(1 + t_0^2)(d_i - d_j) + t_0^2(u_1^i - u_1^j) + 2t_0^2(u_2^i - u_2^j) + \\ & +2t_0(1 + 2t_0 + 5t_0^2)(t_i - t_j) + 2t_0^2(q_i - q_j)|. \end{aligned}$$

As $u_2^l + t_l = \binom{d_i}{2}$ holds, we get:

$$\begin{aligned} d(i,j) \approx \;\; & A\,t_0^2\,|(d_i - d_j)(1 + t_0^2(d_i + d_j)) + t_0^2(u_1^i - u_1^j) + \\ & +2t_0(1 + t_0 + 5t_0^2)(t_i - t_j) + 2t_0^2(q_i - q_j)|. \end{aligned}$$

Cutting the series (1) including terms with t_0^5 we get:

$$\begin{aligned} d(i,j) = \;\; & A\,t_0^2\,|(1 + t_0^2)(d_i - d_j) + t_0^2\,(u_1^i - u_1^j) + 2\,t_0^2\,(u_2^i - u_2^j) \\ & +2\,t_0\,(1 + 2t_0 + 5t_0^2)(t_i - t_j) \\ & +2\,t_0^2\,(q_i - q_j) + 2\,t_0^3\,(u_9^i - u_9^j) + 2\,t_0^3\,(u_{10}^i - u_{10}^j) \\ & +4\,t_0^3\,(u_{11}^i - u_{11}^j) + 8\,t_0^3\,(u_{12}^i - u_{12}^j) + \\ & 12\,t_0^3\,(u_{13}^i - u_{13}^j) + 30\,t_0^3\,(u_{14}^i - u_{14}^j) + 2\,t_0^3\,(p_i - p_j) + \cdots|. \end{aligned}$$

We see that our distance $d(i,j)$ depends on structural parameters characterizing the neighbourhoods of vertices i and j, including graphlet degrees. It seems that only particular graphlets contribute to this measure of vertex similarity. However, higher terms in (1) could be taken into consideration, as well. The choice of t_0 regulates the impact of higher terms. The greater t_0 is, the greater is the length of self-returning walks which significantly contribute to the distance $d(i,j)$.

We shall use a more general result from [9].

Let G be a graph rooted at vertex j. Connected induced subgraphs of G that contain vertex j are called *graphlets* of G at vertex j. These graphlets are also connected rooted graphs that are rooted at j. The number of such graphlets is finite. Suppose that among graphlets there are g mutually non-isomorphic ones and suppose that non-isomorphic graphlets are indexed by integers $0, 1, \ldots, g-1$ in an arbitrary way. As before, the graphlet indexed i is called the *graphlet of type i.* We shall now consider all graphlets that may be present in our graphs, not necessarily those with at most 5 vertices. Let u_i be the number of graphlets of type i. Let further n_i^k be the number of spanning self-returning walks of length k of the graphlet of type i.

The following theorem has been proved in [9]:

Theorem 3. *For any positive integer k and any vertex j of a graph F we have*

$$N_k(j) = \sum_{i=0}^{g-1} n_i^k u_i.$$

Using this theorem we have

$$\mathcal{H}_j(t) = \sum_{k=0}^{\infty} N_k(j) t^k = \sum_{k=0}^{\infty} t^k \sum_{i=0}^{g-1} n_i^k u_i = \sum_{i=0}^{g-1} u_i \sum_{k=0}^{\infty} n_i^k t^k.$$

Let us introduce $S_i(t) = \sum_{k=0}^{\infty} n_i^k t^k$ the generating function for the numbers n_i^k of spanning self-returning walks of length k of the graphlet of type i. Then we have

$$\mathcal{H}_j(t) = \sum_{i=0}^{g-1} u_i S_i(t).$$

Finally we arrive at the following theorem.

Theorem 4. *We have*

$$d(i,j) = A|\sum_{s=0}^{g-1} (u_s^i - u_s^j) S_s(t_0)|.$$

We see that an expression for the generating function $S_s(t)$ would be useful. This will be the subject of future investigations.

6. *Complexity analysis*

In this section we shall analyze the time complexity of a few previously mentioned algorithms in their first phase (determination of similarities between vertices).

As we have seen, GRAAL and H-GRAAL in defining the measure of similarity between vertices of the networks under study use 73-dimensional vector of graphlet degrees. These algorithms are looking for the graphlets up to five vertices in an n-vertex graph. The number of five vertex graphlets is $\binom{n}{5} = \frac{1}{5!}n(n-1)(n-2)(n-3)(n-4)$, so the time complexity of the first phase is $O(n^5)$ (see [15]).

In IsoRank's first phase the measure of similarity $R_{i,j}$ between the vertex i of n-vertex graph G and the vertex j of m-vertex graph H is interpreted as a coordinate of the principal eigenvector of the product $G \times H$. However, it is well-known that the adjacency matrix of $G \times H$ is equal to the Kronecker product of adjacency matrices of graphs G and H and that the principal eigenvector of $G \times H$ is the Kronecker product of principal eigenvectors of graphs G and H (see, for example, [4], pp. 69-70). Hence, principal eigenvectors of graphs G and H can be computed separately. If we for getting each of them apply, for example, the *power method*, the time complexity will be $O(n^2)$.

Since the measure of similarity $R_{i,j}$ is equal to the product of the i-th coordinate of the principal eigenvector of the graph G and the j-th coordinate of the principal eigenvector of the graph H, the time complexity of IsoRank's first phase is $O(n^2)$. Note that in [17] the *power method* has been adopted to the $nm \times nm$ adjacency matrix (i.e. $n^2 \times n^2$ for equal sized vertex sets) what implied the time complexity $O(n^4)$.

It is well known that the time complexity for finding the eigenvalues and the eigenvectors of an matrix is $O(n^3)$. By formulas given in Section 3 graph angles can be quickly computed from eigenvectors. The computation of our measure of similarity requires just finding the eigenvalues and the eigenvectors of corresponding adjacency matrices, so the time complexity in this case will be $O(n^3)$, which is better than in the GRAAL's and H-GRAAL's first phase.

7. *Concluding remarks*

Our treatment of network alignment algorithms is strictly mathematical contrary to the literature of bio-informatics where facts from biology are also taken into account.

We have presented three approaches for determining the measure of similarity between vertices:

- approach based on enumeration of long out-going walks at particular vertices (IsoRank),

- approach based on enumeration of graphlet frequencies at particular vertices (GRAAL),

- approach based on enumeration of self-returning walks at particular vertices (our approach).

Our approach with complexity $O(n^3)$ lies in the middle of the other two. Analysis given in Section 5 shows that the quality of the measure of similarity is comparable with the one in the graphlet approach obtained at a higher computational cost.

The comparison of the approaches can be refined by looking at the cases of failure. In regular graphs the principal eigenvector has all components the same. If both graphs G and H are regular, the measure of similarity for any pair i, j of vertices is constant; hence IsoRank is useless. This happens in our approach for a narrower class of graphs: strongly regular graphs. In strongly regular graphs with same eigenvalues, the angles are also equal and equal for all vertices (see [5], p. 77). However, in other regular graphs angles are powerful. For example, some classes of cubic graphs are characterized up to an isomorphism by eigenvalues and angles (see [5], p. 119). Bad cases with the graphlet approach is more difficult to construct although they probably exist.

In the literature of bio-informatics when comparing network alignment algorithms the stress is led on the quality of obtained alignment in terms of biology. Complexity analysis is not crucial since the number of potentially interesting PPI networks to be aligned is small and it is not much important how long a computer runs.

Since the time complexity of the GRAAL's first phase is greater than the complexity of IsoRank's first phase, there is no doubt that concrete implementation of this algorithm produces better alignments, in fact one of the most complete topological alignments of biological networks (see [10]). It was also reported in [10] that GRAAL finds common subgraphs with more vertices than IsoRank.

It would be interesting and useful to integrate our new measure of similarity into the second phase of some of the mentioned algorithms and implement it for aligning concrete biological networks.

REFERENCES

[1] Blondel V. D., Gajardo A., Heymans M., Senellart P., Van Doren P., *A measure of similarity between graph vertices: applications to synonym extraction and web searching*, SIAM Rev., 46(2004), No. 4, 647–666.

[2] Brin S., Page L., *The Anatomy of Large-Scale Hypertextual Web Search Engine*, Proc. 7th International WWW Conference, 1998.

[3] Cvetković D., *Spectral recognition of graphs*, Yugoslav Journal of Operations Research, **20**(2012), No. **2**, 145–161.

[4] Cvetković D., Doob M., Sachs H., *Spectra of Graphs, Theory and Application*, 3rd edition, Johann Ambrosius Barth Verlag, Heidelberg-Leipzig, 1995.

[5] Cvetković D., Rowlinson P., Simić S. K., *Eigenspaces of Graphs*, Cambridge University Press, Cambridge, 1997.

[6] Golub G. H., Van Loan C. *Matrix computations*, Johns Hopkins University Press, 2006.

[7] Jaccard P., *Étude comparative de la distribution florale dans une portion des alpes et des jura*, Bulletin de la Société Vaudoise des Science Naturelles, **37**(1901), 547–579.

[8] Jovanović I., *Network alignment algorithms*, Proc. XXXIX Symp. on Operational Res. SYMOPIS 2012,(2012), 225–228.

[9] Jovanović I., *Self-returning walks and graphlets*, to appear.

[10] Kuchaiev O., Milenković T., Memiševic V., Hayes W., Pržulj N., *Topological network alignment uncovers biological function and phylogeny*, J R Soc Interface, **7(50)**(2010), 1341–1354.

[11] Kuchaiev O, Pržulj N., *Integrative network alignment reveals large regions of global network similarity in yeast and human*, Bioinformatics, **27(10)**(2011), 1390–1396.

[12] Liao C.-S., Lu K., Baym M., Singh R., Berger B., *IsorankN: spectral methods for global alignment of multiple protein networks*, Bioinformatics, **25**(2009), 253–258.

[13] Memišević V., Pržulj N., *C-GRAAL: Common-neighbors-based global GRAph ALignment of biological networks*, Integr Biol (Camb), **4(7)**(2012), 734–743.

[14] Milenković T., Ng W. L., Hayes W., Pržulj N., *Optimal network alignment with graphlet degree vectors*, Cancer Inform., **9**(2010), 121–137.

[15] Milenković T., Pržulj N., *Uncovering biological network function via graphlet degree signatures*, Cancer Inform., **6**(2008), 257–273.

[16] Papadimitriou C., Steiglitz K. *Combinatorial optimization: algorithms and complexity*, Dover, 1998.

[17] Singh R., Xu J., Berger B., *Pairwise global alignment of protein interaction networks by matching neighborhood topology*, Research in Computational Molecular Biology, Springer, (2007), 16–31.

[18] Xuan Q., Yu L., Du F., Wu T.-J., *A review on node-matching between networks*, New Frontiers in Graph Theory, Ed. Y. Zhang, Intex, Rijeka, 2012, 153–167.

[19] Wei T. H., The algebraic foundations of ranking theory, Thesis, Cambridge, 1952.

[20] Zaslavskiy M., Bach F., Vert J.-P., *Global alignment of protein-protein interaction networks by graph matching methods*, Bioinformatics, **25**(2009), i259–i267.

Mathematical Institute SANU
Knez Mihailova 36
11001 Belgrade
Serbia
e-mail: ecvetkod@etf.rs

School of Computing
Union University
Knez Mihailova 6
11001 Belgrade
Serbia
e-mail: irenaire@gmail.com